

Offset-Free TAGE-SC: Conditional-Branch-Level Prediction for Wide Fetch Frontends

Yashwant Kumar Balivada*

Department of Electrical and Computer Engineering
Texas A&M University
College Station, USA

Sairam Viswanathan Susarla*

Department of Electrical and Computer Engineering
Texas A&M University
College Station, USA

Abstract—Wide-fetch branch prediction must improve accuracy without overspending latency, energy, or storage. We propose OF-TAGE-SC, an offset-free TAGE-SC predictor that organizes prediction around conditional-branch order within a fetch block rather than branch offsets. Using the CBP-NG aggregate scoring methodology over 168 public training traces, the submitted OF-TAGE-SC configuration achieves an aggregate VFS of 0.9374. Although GShare-Ahead obtains the highest aggregate VFS of 0.9736, OF-TAGE-SC provides better final-prediction accuracy than GShare-Ahead, reducing MPKI from 6.93 to 5.97 while using much less modeled storage. Relative to TAGE, OF-TAGE-SC improves aggregate VFS by 7.4%, increases IPC_{cbp} by 26.7%, and reduces EPI_{cbp} by 28.8%, while increasing MPKI from 5.49 to 5.97. A follow-up design-space sweep with $MAXINST=64$ shows additional headroom, reaching a best observed aggregate VFS (Voltage Frequency-scaled Speedup) of 0.9525 at $N = 8$. These results show that conditional-branch-order organization improves the hardware-efficiency tradeoff, even though the global VFS optimum favors a high-throughput, low-energy baseline.

Index Terms—branch prediction, TAGE-SC, CBP-NG, VFS, energy, storage, wide fetch

I. INTRODUCTION

Modern frontends need accurate branch prediction, but CBP-NG[1] makes clear that accuracy alone is not enough. A predictor that lowers MPKI can still lose if it increases table activity, predictor latency, or storage. Conversely, a very cheap predictor can achieve attractive energy and timing but collapse on difficult control-flow traces. The useful design point is therefore the one that balances MPKI, IPC/CPI, EPI, latency, and storage.

This paper targets a specific inefficiency in wide-fetch predictors: offset-level organization. A fetch block can contain several conditional branches, yet an offset-aware predictor may allocate prediction effort across many possible instruction positions. This increases table reads, tag comparisons, and selection logic even when only a few conditional branches drive control flow. Our design instead predicts by conditional-branch order. The predictor asks: what are the next few conditional branches in this block likely to do? It does not require a separate prediction identity for every fetch offset.

Offset-Free (OF) TAGE-SC combines a fast, low-latency GShare-style first-stage predictor with a slower, more accurate

branch-order TAGE-SC predictor. The design keeps TAGE’s long-history strength, adds a lightweight statistical corrector, and uses a branch-order capacity parameter N to control how many conditional branches are tracked in a block. Our final design improves VFS while remaining close to TAGE in storage and far below GShare-Ahead’s table footprint.

A. Related Work

TAGE uses tagged tables with geometric history lengths and remains a strong low-MPKI baseline [2]. TAGE-SC adds statistical correction to reduce weak or biased TAGE errors [3]. GShare uses XOR-indexed counters and is attractive for low-cost prediction. Ahead/pipelined predictors improve frontend progress by predicting future blocks early [5], [6]. Perceptron predictors learn long-history correlations through weighted sums and provide a useful neural-style baseline [4].

II. DESIGN OVERVIEW

OF-TAGE rethinks the conventional TAGE prediction process by shifting from offset-oriented branch prediction to an instruction-window-oriented prediction model. Unlike the original TAGE design, where each branch independently performs tag lookups and comparisons across multiple predictor tables, OF-TAGE constrains prediction generation to a fixed set of conditional-branch positions within an instruction window. This contrast between conventional offset-based storage and OF-TAGE’s conditional-branch-position-based storage is illustrated in Fig. 1. This alternative organization reduces the number of offset-specific prediction entries that must be considered and focuses predictor capacity on the ordered conditional branches that actually appear in the block. In the submitted OF-TAGE-SC configuration, this tradeoff improves IPC_{cbp} from 5.77 to 7.31 and reduces EPI_{cbp} from 1265.5 fJ/inst. to 901.2 fJ/inst. relative to TAGE, while keeping the same reported P1/P2 latency class. The cost is a modest increase in MPKI from 5.49 to 5.97, but the aggregate VFS still improves from 0.8728 to 0.9374. Thus, the benefit of the offset-free organization is not purely accuracy; it improves the TAGE-style hardware-efficiency point by trading a small final-misprediction increase for higher prediction throughput and lower predictor energy.

At a hardware level, OF-TAGE-SC follows a common two-speed predictor organization. A fast first-stage predictor

*Both authors contributed equally to this work.

provides an early direction estimate for the frontend. A slower, more accurate second-stage predictor refines that decision using larger tables and longer histories. In the CBP-NG model these correspond to the scored first and second predictor stages, but the design motivation is architectural: keep the early predictor simple and reserve the expensive logic for cases where it improves final prediction quality.

The first-stage predictor is a banked GShare-style structure. It is not intended to be the primary accuracy engine; instead, its role is to provide quick branch-order predictions with low table cost and fast adaptation behavior. The second-stage predictor is a branch-order TAGE-SC design built on top of our OF-TAGE (Offset-Free TAGE) predictor. Similar to conventional TAGE predictors, its prediction components use tagged geometric history tables; however, unlike offset-oriented block predictors, the tables in OF-TAGE are organized using the rank of a conditional branch within a prediction block rather than its fetch offset. This branch-level organization forms the key distinction between OF-TAGE-SC and traditional offset-based block predictors.

The statistical corrector uses three low-latency contexts: branch-PC bias, taken conditional-branch history, and global path history. We explored additional contexts, including TAGE-confidence input, but rejected them when they introduced serial dependence into the accurate predictor path. This is important: several features reduced MPKI in isolation, but not all of them improved VFS once timing and energy were considered.

III. DETAILED PREDICTOR OPERATION

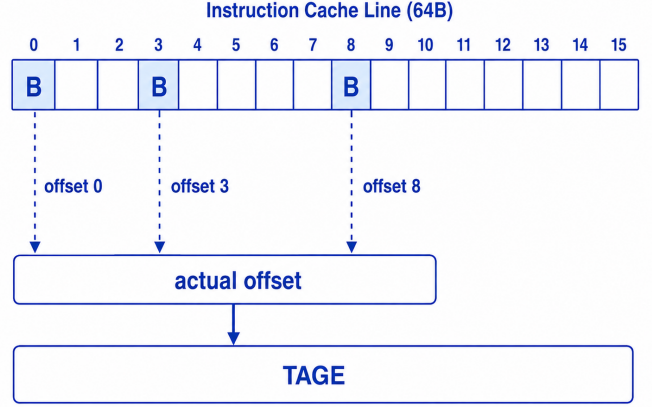
For each fetch block, the predictor maintains an ordered branch position rather than an offset position. The fast predictor uses global history and PC-derived hashing to read a small banked table and generate early branch-order predictions. This gives the frontend a quick candidate direction while the accurate predictor performs larger table lookups.

The accurate predictor reads tagged geometric-history tables and selects the longest matching provider in the usual TAGE style. Alternate predictions and usefulness/hysteresis state stabilize allocation. The difference is that the candidate being predicted is the i th conditional branch in a block, not the instruction at a particular byte or instruction offset. This allows branch capacity to be controlled by N . If N is too small, the predictor misses useful branch-level coverage; if N is too large, extra banks and table activity can dominate the score.

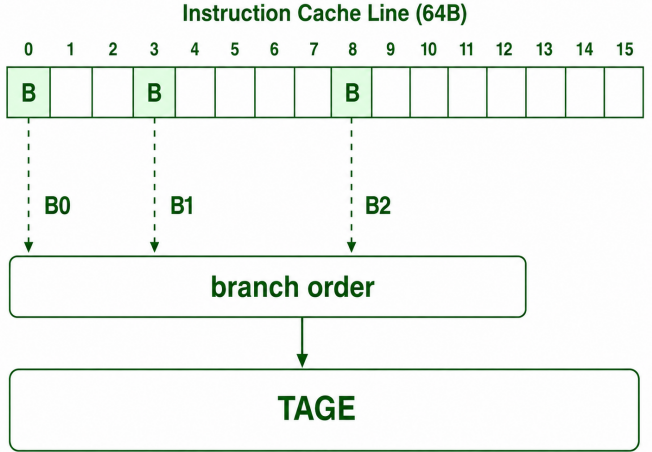
The SC override is applied only when its confidence is high, as shown in Fig. 2. The three final SC features are independent enough to avoid a long serial chain through TAGE. This kept the design in the same scored second-stage latency class as TAGE-like baselines while improving the global VFS tradeoff.

IV. METHODOLOGY

We evaluate the predictors on the public CBP-NG training traces using the HARCOM-based CBP-NG simulator. The simulator logs raw execution and predictor quantities,



(a) Traditional Offset-Based TAGE



(b) Offset-Free TAGE

Fig. 1: Offset-based vs. conditional-branch-order TAGE branch storage

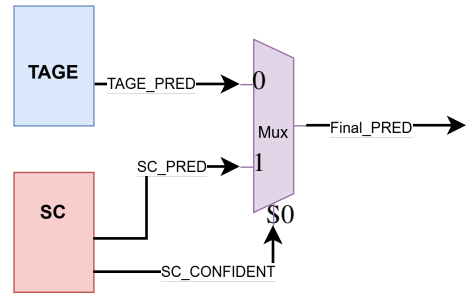


Fig. 2: OF-TAGE-SC Architecture

including correct-path instructions, predictions, extra cycles, short mispredictions, long/final mispredictions, block-ending short mispredictions, predictor energy, and first/second-stage predictor latency. From these quantities, we report the three figures of merit used by the CBP-NG VFS score: prediction throughput IPC_{cbp} , long-misprediction penalty CPI_{cbp} , and predictor energy EPI_{cbp} .

Following the CBP-NG scoring methodology, IPC_{cbp} is ag-

gregated across traces using a harmonic mean, while CPI_{cbp} and EPI_{cbp} are aggregated using arithmetic means. VFS is then computed once from these aggregate figures of merit. We therefore do not compute or interpret VFS for individual traces or trace classes. We additionally report MPKI as a diagnostic accuracy metric, computed from final/long mispredictions per thousand correct-path instructions. Short mispredictions are not counted as MPKI because they represent disagreements between the first and second predictions rather than final wrong-direction predictions.

The metrics interact rather than rank independently. MPKI captures final prediction accuracy pressure. IPC_{cbp} captures frontend prediction throughput. CPI_{cbp} captures the cycle penalty from long mispredictions. EPI_{cbp} captures dynamic predictor energy per correct-path instruction. A predictor can therefore be attractive on one axis but lose aggregate VFS if it pays too much on another axis. For example, a high-throughput predictor may still lose score if its long-misprediction penalty or energy is too high, while a low-energy predictor may lose if its final prediction accuracy is weak.

The trace set spans several trace-name-derived workload classes: SPEC-like integer and floating-point traces, JVM/Java/DaCapo/Renaissance/SPECjbb, Web/JavaScript/Node, compression, infrastructure/server traces, solver and geometry traces, graph workloads, database workloads, and FP/media traces. These classes are used only for diagnostic interpretation of the underlying metrics; all scored averages and VFS values are computed over the full evaluated trace set.

V. EXPERIMENTAL RESULTS AND ANALYSIS

Table I summarizes the aggregate score, final-misprediction behavior, latency class, and modeled storage cost of each predictor. Table II reports the three aggregate CBP-NG figures of merit used to compute VFS: IPC_{cbp} , CPI_{cbp} , and EPI_{cbp} . MAXINST denotes the maximum instruction window size considered per fetch block. The OF-TAGE-SC row corresponds to the submitted configuration, which uses $N = 4$ and MAXINST=24. GShare-Ahead obtains the highest aggregate VFS, primarily because its high IPC_{cbp} and low EPI_{cbp} in Table II compensate for its weaker MPKI in Table I. OF-TAGE-SC provides the strongest TAGE-style design point in this comparison. Relative to TAGE, it improves aggregate VFS from 0.8728 to 0.9374, increases IPC_{cbp} from 5.77 to 7.31, and reduces EPI_{cbp} from 1265.5 fJ/inst. to 901.2 fJ/inst. This comes with an increase in MPKI from 5.49 to 5.97 and CPI_{cbp} from 0.0549 to 0.0597. Thus, OF-TAGE-SC should be interpreted as a better hardware-efficiency tradeoff than TAGE, not as the globally highest-VFS predictor among all baselines.

Together, Tables I and II highlight why MPKI alone is insufficient. TAGE has the best MPKI and the lowest CPI_{cbp} , but its low IPC_{cbp} and high EPI_{cbp} reduce its aggregate VFS. GShare-Ahead shows the opposite behavior: it has weaker MPKI than TAGE and OF-TAGE-SC, but its high prediction throughput and low predictor energy make it the highest-VFS design. OF-TAGE improves IPC_{cbp} and EPI_{cbp} relative

TABLE I: Aggregate Score, Accuracy, Latency, and Storage

Predictor	MPKI	VFS	P1/P2 Lat	KiB
GShare-Ahead	6.93	0.9736	1/1	128.0
OF-TAGE-SC	5.97	0.9374	1/2	36.5
OF-TAGE	7.20	0.9287	1/2	38.0
TAGE	5.49	0.8728	1/2	35.0
Hashed Perc.	5.64	0.8551	1/2	28.9
Perceptron	7.16	0.8482	0/2	28.9
GShare	7.43	0.8430	1/2	8.0

TABLE II: Aggregate VFS Input Figures of Merit

Predictor	IPC_{cbp}	CPI_{cbp}	EPI_{cbp}
GShare-Ahead	9.20	0.0624	243.1
OF-TAGE-SC	7.31	0.0597	901.2
OF-TAGE	7.85	0.0720	672.2
TAGE	5.77	0.0549	1265.5
Hashed Perc.	5.71	0.0564	1891.6
Perceptron	5.83	0.0716	621.9
GShare	5.86	0.0743	665.2

to TAGE, but its higher MPKI and CPI_{cbp} keep it below OF-TAGE-SC in aggregate VFS. The hashed perceptron has competitive MPKI, but its very high EPI_{cbp} limits its score. These results show that OF-TAGE-SC improves the TAGE-family tradeoff by moving closer to the throughput and energy region of cheaper predictors while retaining accuracy closer to TAGE.

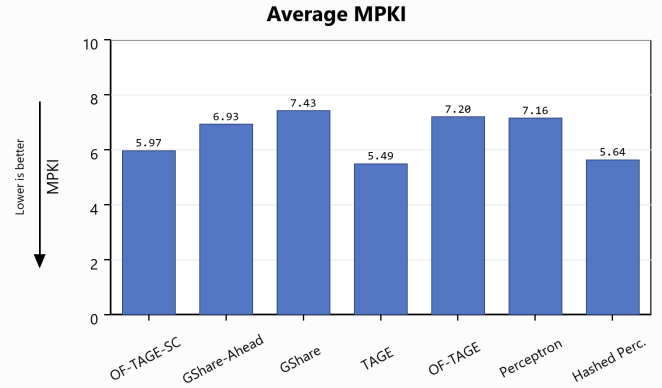


Fig. 3: Aggregate MPKI across predictors

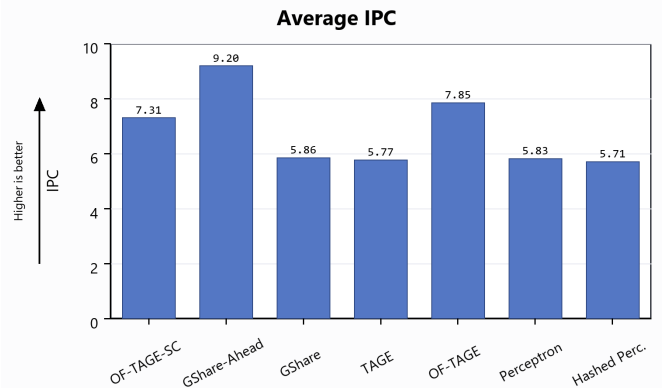


Fig. 4: Harmonic-mean IPC_{cbp} across predictors

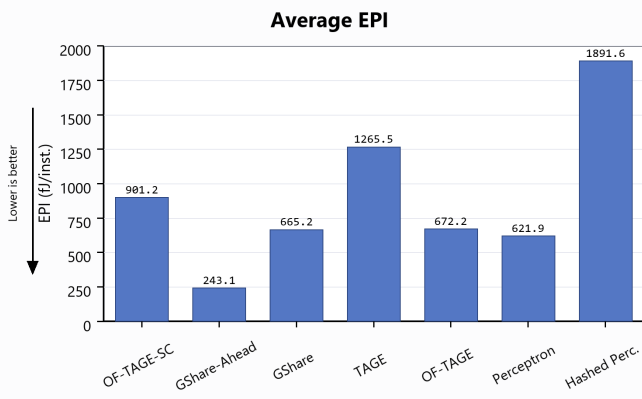


Fig. 5: Arithmetic-mean EPI_{cbp} across predictors

A. Trace-Class Behavior

Fig. 6 and Fig. 7 show diagnostic trace-class behavior for EPI_{cbp} and IPC_{cbp} . These plots are not used to compute VFS and should not be interpreted as per-class VFS results. Instead, they help explain which underlying metrics drive the aggregate score reported in Tables I and II.

The main observation is that the predictors occupy different regions of the design space. GShare-Ahead demonstrates the value of high prediction throughput and low predictor energy, which gives it the highest aggregate VFS in this comparison. OF-TAGE-SC provides the strongest TAGE-style design point: relative to TAGE, it substantially improves prediction throughput and predictor energy while keeping MPKI close to the best final-prediction baselines. This makes OF-TAGE-SC a storage-comparable improvement that moves the design toward the efficiency region of simpler high-throughput predictors without giving up the accuracy advantages of long-history prediction.

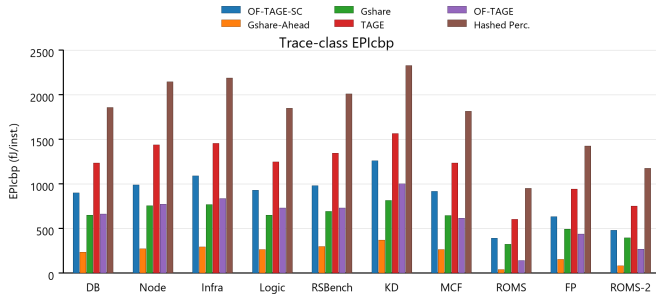


Fig. 6: Trace-class EPI_{cbp} comparison

On easier trace classes, several predictors have low final-misprediction pressure, so differences in IPC_{cbp} and EPI_{cbp} become more important. On harder control-flow classes, final prediction accuracy and CPI_{cbp} become more visible in the aggregate metrics. OF-TAGE-SC benefits when its branch-order organization reduces predictor cost and improves throughput relative to TAGE without causing a large increase in long mispredictions. It loses ground when the throughput and energy advantage of GShare-Ahead dominates.

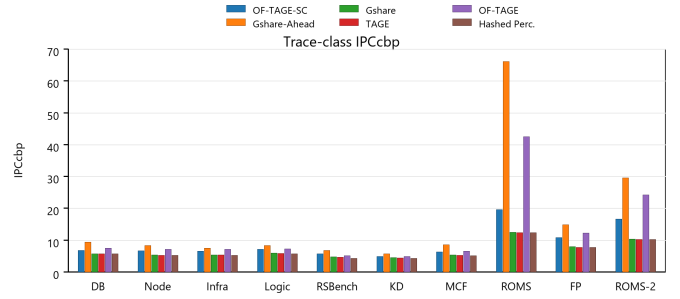


Fig. 7: Trace-class IPC_{cbp} comparison

B. Sweep of Branch-Order Capacity

The submitted OF-TAGE-SC configuration uses $N = 4$ and $MAXINST=24$. We performed an additional design-space sweep to test whether larger branch-order capacity and a larger instruction window could further improve the aggregate score. This sweep uses $MAXINST=64$ and varies N , the number of conditional-branch positions tracked within a block.

The follow-up sweep shows additional headroom. At $MAXINST=64$, aggregate VFS rises from 0.9384 at $N = 3$ to 0.9490 at $N = 4$, then continues to improve gradually, reaching the best observed score of 0.9525 at $N = 8$. The larger settings, $N = 9$, $N = 10$, and $N = 11$, still fall below the $N = 8$ sweep optimum, with VFS values of 0.9424, 0.9436, and 0.9430, respectively. This suggests that larger branch-order capacity is useful, but the best point depends on the balance between coverage, throughput, and energy rather than on maximizing N alone.

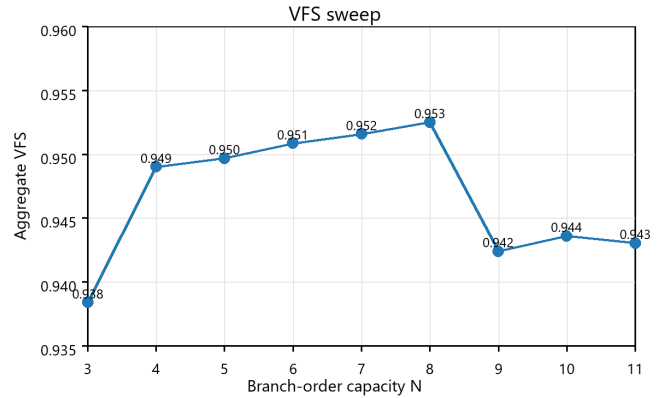


Fig. 8: Aggregate VFS sweep for OF-TAGE-SC at $MAXINST=64$

The sweep supports a useful-capacity interpretation. From $N = 3$ to $N = 8$, the added branch-order coverage improves IPC_{cbp} and slightly reduces long-misprediction pressure. At $N = 8$, the design reaches $IPC_{cbp} = 7.81$, $CPI_{cbp} = 0.0585$, $MPKI = 5.85$, and $EPI_{cbp} = 942.0$ fj/inst. Increasing beyond $N = 8$ with the additional branch-position coverage no longer outperforms the $N = 8$ balance of throughput, accuracy, and energy. The sweep therefore suggests that branch-level capacity should be tuned to the useful branch-density point rather than maximized blindly.

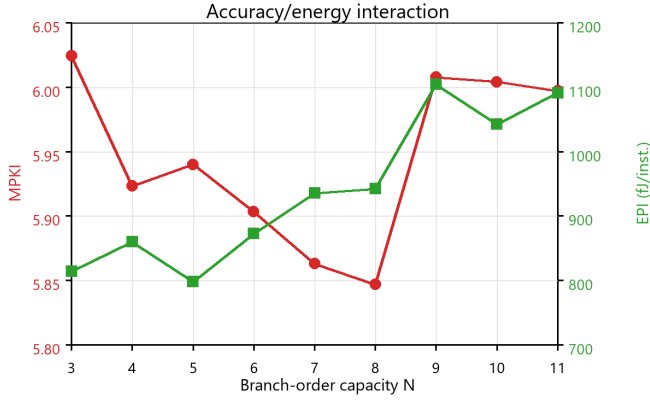


Fig. 9: Accuracy/energy interaction for the N sweep

Fig. 9 provides more context for the trend observed in the N sweep by separating the accuracy and energy components behind the aggregate score. Increasing N expands the number of conditional-branch positions that can be represented within a block, which can improve coverage and reduce long-misprediction pressure. At the same time, the added capacity changes the predictor activity profile and increases EPI_{cbp} . The figure shows that the best design point is therefore not determined by accuracy alone or energy alone, but by how these effects combine.

C. Storage Budget

Before interpreting the final ranking, storage matters. In the main comparison, the submitted OF-TAGE-SC configuration uses 36.5 KiB of modeled persistent predictor-table state, only 1.5 KiB above TAGE and about $3.5\times$ smaller than GShare-Ahead. This places OF-TAGE-SC near the TAGE storage region while achieving a substantially higher aggregate VFS than both TAGE and OF-TAGE. The result shows that conditional-branch-order organization improves the efficiency point by spending prediction capacity on branch order rather than fetch offset.

VI. DISCUSSION

The CBP-NG/HARCOM results show that OF-TAGE-SC improves the TAGE-style design point by organizing prediction around conditional-branch order instead of fetch offset. Relative to TAGE, the submitted OF-TAGE-SC configuration raises aggregate VFS from 0.8728 to 0.9374, increases IPC_{cbp} from 5.77 to 7.31, and reduces EPI_{cbp} from 1265.5 fj/inst. to 901.2 fj/inst., while remaining close to TAGE in modeled storage. This demonstrates that branch-order prediction improves frontend progress and predictor energy efficiency while preserving the benefits of long-history TAGE-style prediction.

The broader comparison also highlights the strength of OF-TAGE-SC as a balanced wide-fetch predictor. GShare-Ahead represents a high-throughput, low-energy reference point, while OF-TAGE-SC provides stronger final-prediction accuracy than GShare-Ahead and substantially smaller modeled storage. The MAXINST=64 sweep further supports the design principle, reaching a best observed aggregate VFS of

0.9525 at $N = 8$. This shows that branch-order capacity is a useful tunable parameter for improving the accuracy, throughput, and energy balance.

VII. CONCLUSION

OF-TAGE-SC reorganizes TAGE-SC around conditional-branch order rather than fetch offset. This approach focuses predictor capacity on the ordered conditional branches within a fetch block and improves the hardware-efficiency tradeoff.

Under the aggregate CBP-NG scoring methodology, OF-TAGE-SC achieves a submitted aggregate VFS of 0.9374, improving over TAGE by 7.4%, increasing IPC_{cbp} by 26.7%, and reducing EPI_{cbp} by 28.8%. The MAXINST=64 sweep reaches a best observed VFS of 0.9525 at $N = 8$, showing additional scalability of the branch-order organization. Overall, OF-TAGE-SC demonstrates that branch-order-aware prediction is an effective strategy for balancing accuracy, throughput, energy, latency, and storage in wide-fetch frontends.

REFERENCES

- [1] “CBP-NG: Next-Generation Championship Branch Prediction Infrastructure,” public branch-prediction evaluation infrastructure.
- [2] A. Seznec and P. Michaud, “A case for (partially) tagged geometric history length branch prediction,” *Journal of Instruction-Level Parallelism*, 2006.
- [3] A. Seznec, “TAGE-SC-L branch predictors again,” in *Championship Branch Prediction Workshop*, 2016.
- [4] D. A. Jimenez and C. Lin, “Neural methods for dynamic branch prediction,” *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 369–397, 2002.
- [5] A. Seznec, S. Jourdan, P. Sainrat, and P. Michaud, “Multiple-block ahead branch predictors,” in *Proceedings of ASPLOS*, 1996.
- [6] A. Seznec and A. Fraboulet, “Effective ahead pipelining of instruction block address generation,” in *Proceedings of ISCA*, 2003.